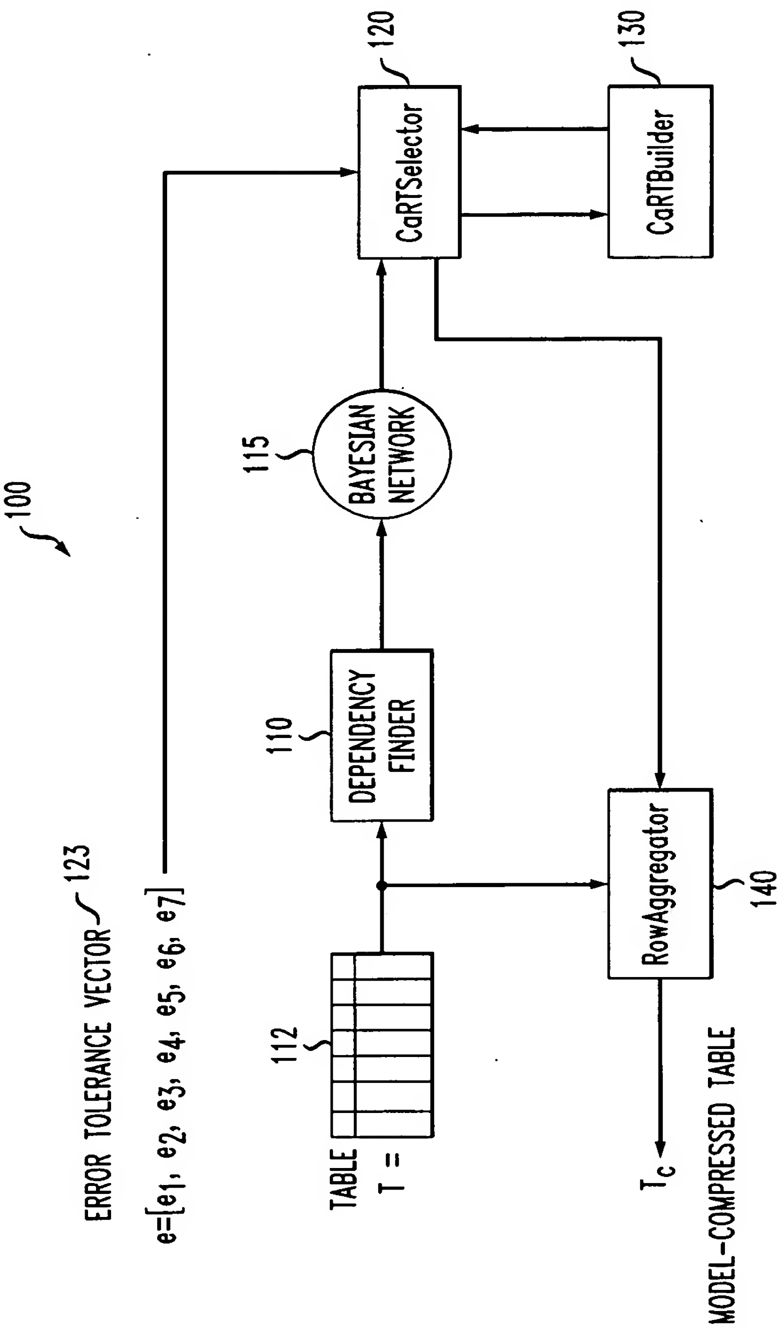

100
ERROR TOLERANCE VECTOR 123
e=[e1, e2, e3, e4, e5, e6, e7]
TABLE
T =
112
110
DEPENDENCY FINDER
115
BAYESIAN NETWORK
120
CaRTSelector
130
CaRTBuilder
RowAggregator
140
Tc
MODEL-COMPRESSED TABLE

FIG. 1

FIG. 2

The Greedy CaRT Selection Algorithm

procedure Greedy $(T(X), \bar{e}, G, \theta)$

Input: n-attribute table T and n-vector of error tolerances $\bar{e}$; Bayesian network G on the set of attributes X and threshold θ on the relative benefit for selecting a CaRT predictor.

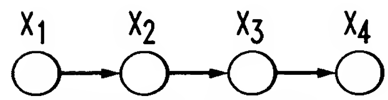
Output: A set of materialized (predicted) attributes X_{mat} ($X_{pred} = X - X_{mat}$) and a CaRT predictor for each $X_i \in X_{pred}$.

begin

1. $X_{mat} := X_{pred} := \Phi$
2. let $< X_1, X_2, \ldots, X_n >$ be the attributes in X sorted in *topological order* of G
3. for i :=1,...,n
4. if $\Pi(X_i) = \Phi$ then $X_{mat} := X_{mat} \cup \{X_i\}$ /* X_i must be materialized if it has no parents in G */
5. else
6. M := BuildCaRT ($X_{mat} \rightarrow X_i, e_i$)
7. if (MaterCost (X_i) / PredCost ($X_{mat} \rightarrow X_i$) > θ) then $X_{pred} := X_{pred} \cup \{X_i\}$
8. else $X_{mat} := X_{mat} \cup \{X_i\}$
9. end
10. end

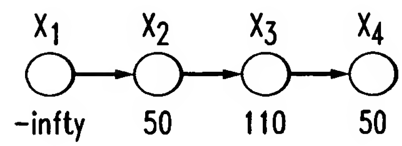
end

FIG. 3A



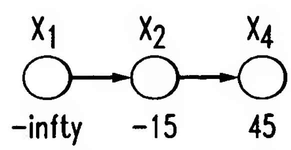
BAYESIAN NETWORK G

FIG. 3B



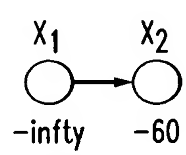
(b) Gtemp (FIRST ITERATION)

FIG. 3C



Gtemp (SECOND ITERATION)

FIG. 3D



Gtemp (THIRD ITERATION)

FIG. 4

The MaxIndependentSet CaRT Selection Algorithm

procedure MaxIndependentSet $(T(X), \bar{e}, G, \text{neighborhood}())$
Input: n-attribute table T and n-vector of error tolerances $\bar{e}$;
Bayesian network G on the set of attributes X and function neighborhood () defining the "predictive neighborhood" of a node X_i in G (e.g., $\Pi(X_1)$ or $\beta(X_i)$).
Output: A set of materialized (predicted) attributes X_{mat} ($X_{pred} = X - X_{mat}$) and a CaRT predictor PRED $(X_i) \rightarrow X_i$ for each $X_i \in X_{pred}$.
begin
1. $X_{mat} := X$, $X_{pred} := \phi$
2. PRED $(X_i) := \phi$ for all $X_i \in X$, improve := true
3. while (improve $\neq$ false) do
4. for each $X_i \in X_{mat}$
5. mater_neighbors (X_i) := $(X_{mat} \cap \text{neighborhood}(X_i)) \cup \{\text{PRED}(X) : X \in \text{neighborhood}(X_i), X \in X_{pred}\} - \{X_i\}$
6. M := BuildCaRT (Mater_neighbors $(X_i) \rightarrow X_i$, e_i)
7. let PRED $(X_i) \subseteq$ mater_neighbors (X_i) be the set of predictor attributes used in M
8. cost_change$_i$:=0
9. for each $X_j \in X_{pred}$ such that $X_i \in$ PRED (X_j)
10. NEW_PRED$_i$ (X_j) := PRED$(X_j) - \{X_i\} \cup$ PRED (X_i)
11. M :=BuildCaRT (NEW_PRED$_i$ $(X_j) \rightarrow X_j$, e_j
12. set NEW_PRED$_i$ (X_j) to the (sub) set of predictor attributes used in M
13. cost_change$_i$:= cost_change$_i$ + (PredCost (PRED $(X_j) \rightarrow X_j$) - PredCost (NEW_PRED$_i$ $(X_j) \rightarrow X_j$))
14. end
15. end

FIG. 4 (cont)

```
    build an undirected, node-weighted graph G_temp = (X_mat, E_temp) on the current set of materialized
    attributes X_mat, where:
        (a) E_temp := { (X,Y) : for all pairs X, Y ∈ X_pred } ∪
                                  { (X_i, Y) : for all Y ∈ X_mat }
        (b) weight (X_i) := MaterCost (X_i) -PredCost (PRED(X_i)
            → X_i) +cost_change_i  for each X_i ∈ X_mat
    S := FindWMIS (G_temp)     /* select (approximate) maximum
    weight independent set in G_temp
                                     as "maximum-benefit" subset of
                                     predicted attributes */
    if (Σ_{X∈S} weight (X) ≤ 0) then improve := false
    else/* update X_mat, X_pred , and the chosen CaRT predictors */
        for each X_j ∈ X_pred
            if (PRED (X_j) ∩ S = {X_i}) then PRED (X_j) :=
            NEW_PRED_i (X_j)
        end
        X_mat := X_mat − S,  X_pred := X_pred ∪ S
    end
end /* while */
end
```

FIG. 5

Algorithm for Estimating Lower Bound on Subtree Cost

```
procedure LowerBound (N, e_i, b)
Input:  Leaf N for which lower bound on subtree cost is to be
        computed; error tolerance e_i for attribute X_i; bound b
        on the maximum number of internal nodes in subtree
        rooted at N.
Output: Lower bound L(N) on cost of subtree rooted at N.
begin
    for i := to r
        minOut [i, 0] :=i
    for J := 1 to b + 1
        minOut [0, j] :=0
    1 :=0
    for i := 1 to r
        while x_i - x_1+1 > 2 ei
        1 :=1 = 1
    for j := 1 to b + 1
        minOut [i,j] := min {minOut[i - 1,j] + 1, minOut [1,j-1]
    end
    L(N) := ∞
    for J := 0 to b
        L(N) :=min {L(N) , 2j + 1 + j log (|X_i|) + (j + 1 + minOut
        (r, j+1)) log (|dom(X_i)|)}
    L(N) := min {L(N) , 2b + 3 + (b + 1) log (|X_i|) + (b +2) log
    (|dom(X_i) |) }
    return L (N)
end
```

FIG. 6

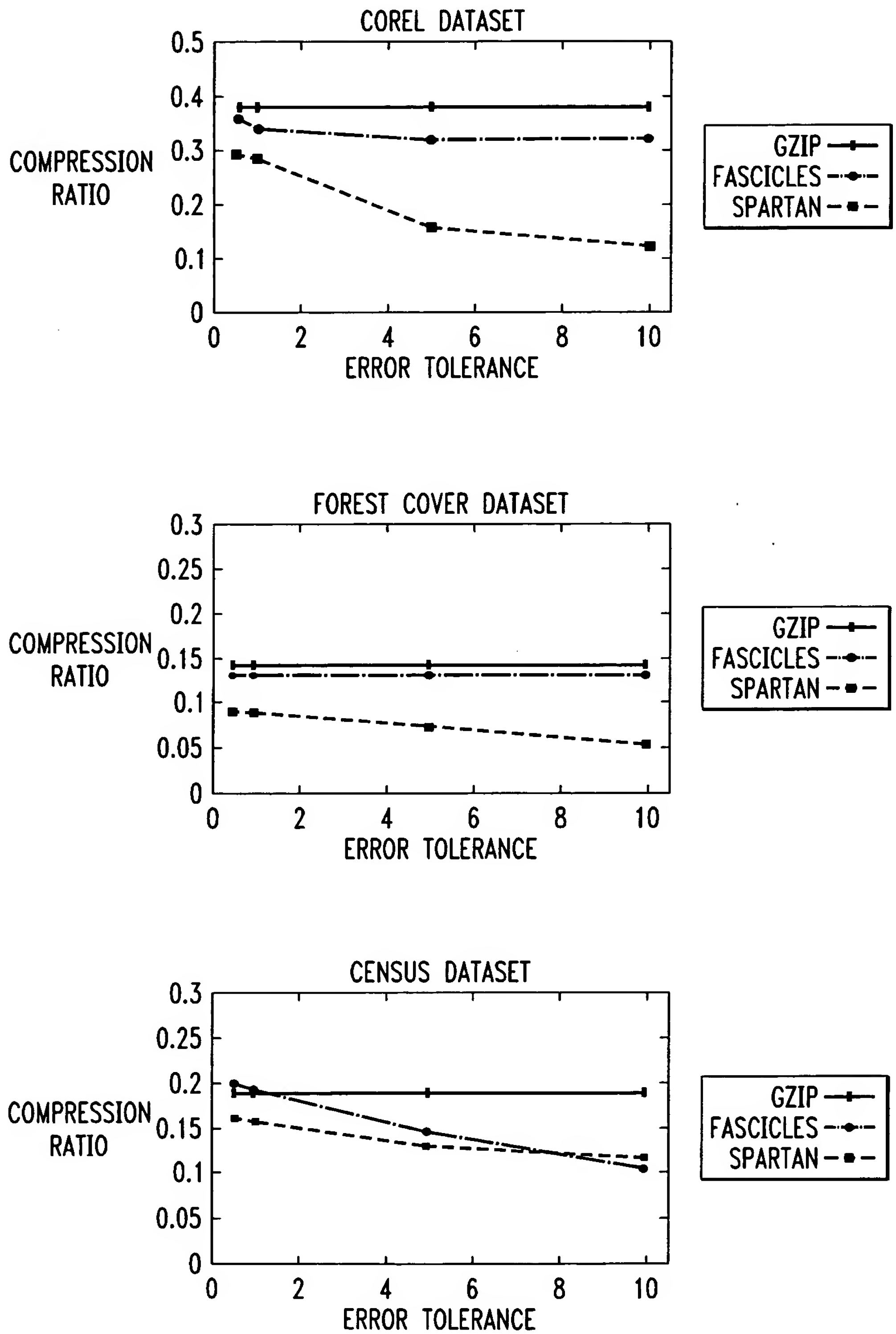

COREL DATASET
COMPRESSION RATIO
ERROR TOLERANCE
GZIP
FASCICLES
SPARTAN
FOREST COVER DATASET
COMPRESSION RATIO
ERROR TOLERANCE
GZIP
FASCICLES
SPARTAN
CENSUS DATASET
COMPRESSION RATIO
ERROR TOLERANCE
GZIP
FASCICLES
SPARTAN

FIG. 7A

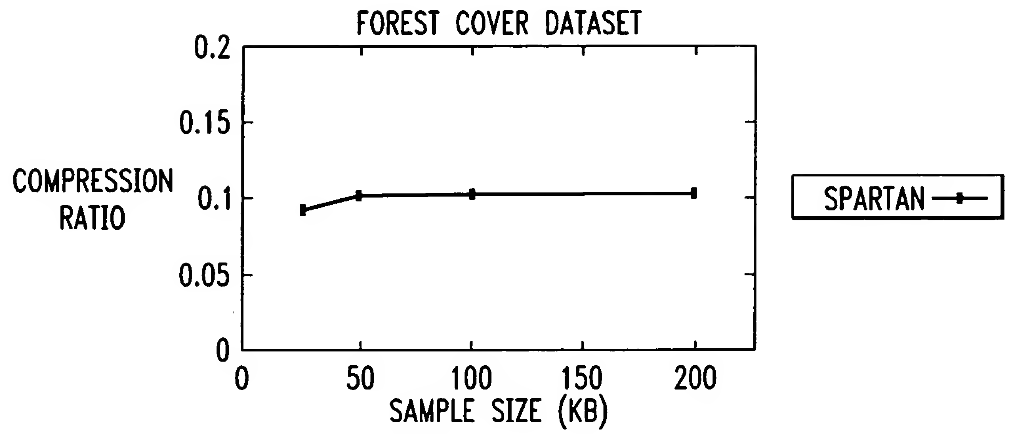


FIG. 7B

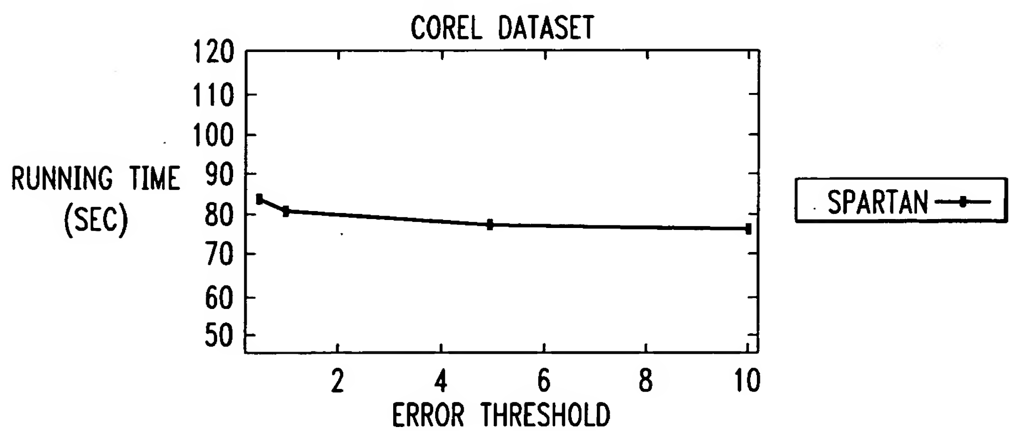


FIG. 7C